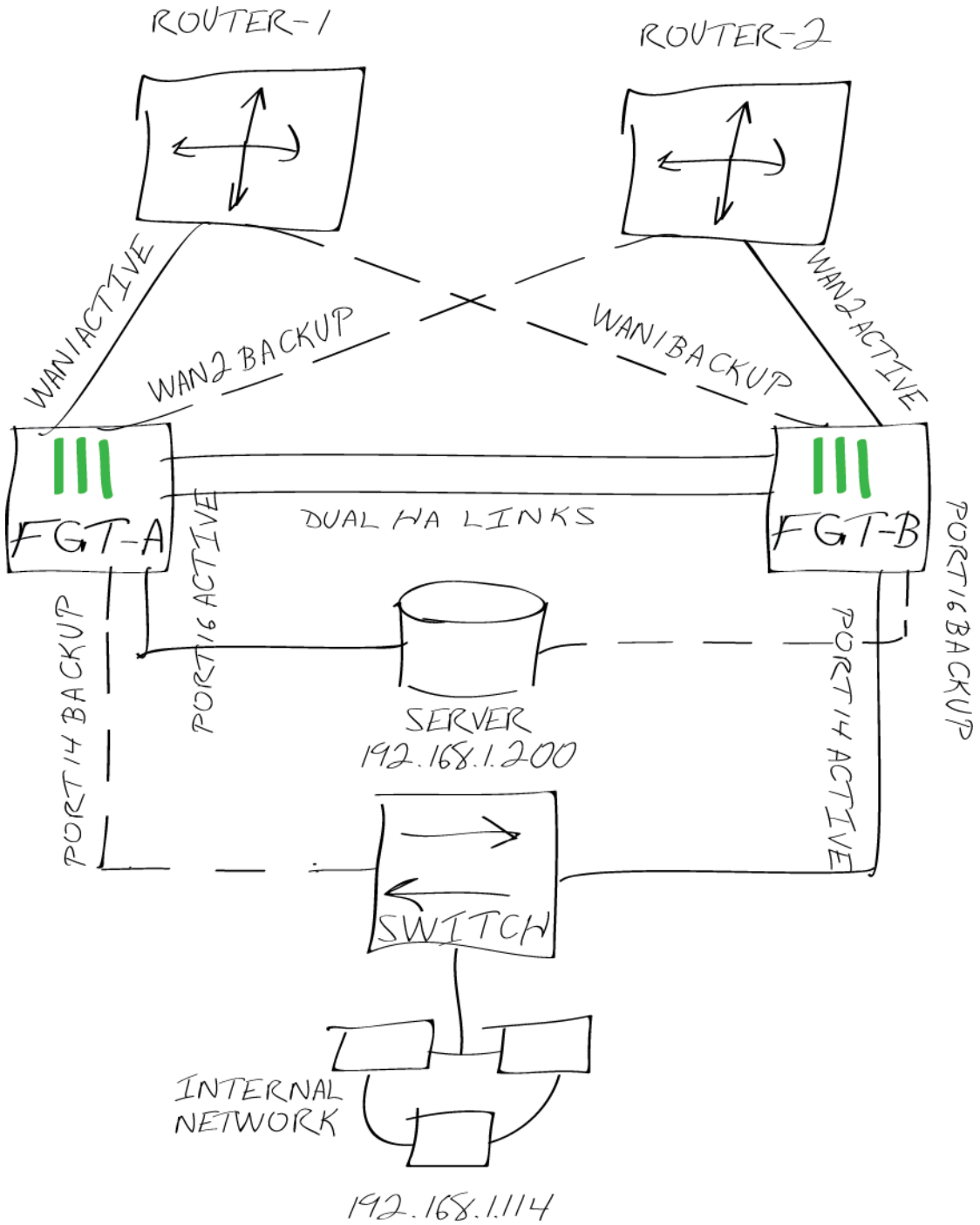
Fortinet Redundant UTM Protocol (FRUP)

1. Network topologies



2. FRUP requirement and assumptions

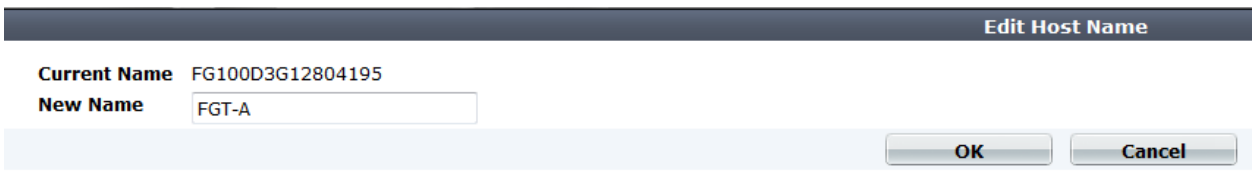
1. Maximum 2 units, FGT-100D model only (for now), will run in this type of cluster
2. Use dual HA links **HA1** and **HA2** for redundant HA connectivity between the cluster members
3. Use dual gateways (routers above) such that:
    - Device-1 has default active connection to router-1 and default backup connection to router-2
    - Device-2 has the opposite
4. Design the lower network in the similar fashion as WAN connectivity:
    - Each device (server, AP, etc) will have a default active connection to one of the FortiGates and a default backup connection to the other one
    - Default active & backup should be alternated to balance the traffic
5. HA using FGCP with a full mesh configuration requires 2 pairs of L2 switches. HA using FRUP collapses the L2 switching into the FortiGate 100D model. Client hosts are dual-homed on the FRUP FortiGate units and assumed to have two NICs or use an intermediary consolidation switch device.
6. Follows most HA settings but mode is “standalone”
7. FortiGate interfaces use virtual IP addresses and pseudo-MAC physical addresses, all devices continue to send to the same IP/Mac and don't need to re-learn after the failover
8. Both FortiGate units will handle and process traffic
9. Backup lines are normally administratively down and de-energized
10. Sessions and configuration are synchronized in FRUP cluster
11. From FMG perspective it is virtually unchanged from regular HA cluster

3. Configuring FGT-A

Refer to “Using two ISPs for redundant Internet connections” recipe.

Set WAN1 and WAN2 interfaces to use static addressing mode and set both static routes to same priority and distance.

Go to **System > Dashboard > Status** and under “System Information” widget, change the host name to **FGT-A**



From CLI set the following:

```
FGT-A # config system ha
   set hbdev "ha1" 50 "ha2" 100
   set override disable
   set priority 255
   set frup enable
      config frup-settings
```

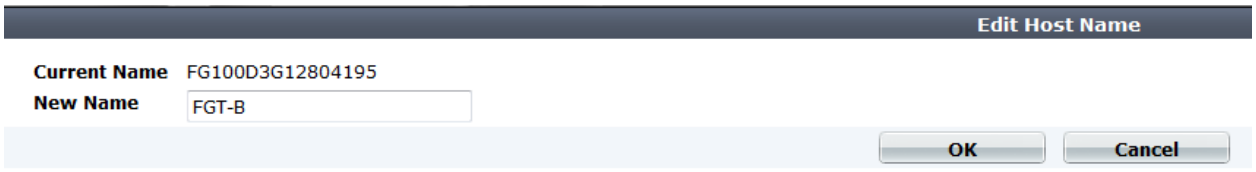
```
        set active-interface "wan1"
        set active-switch-port 16
        set backup-interface "wan2"
    end
end
```

```
FGT-A # sh sys ha
config system ha
    set hbdev "ha1" 50 "ha2" 100
    set override disable
    set priority 255
    set frup enable
        config frup-settings
            set active-interface "wan1"
            set active-switch-port 16
            set backup-interface "wan2"
        end
end

FGT-A #
```

4. Configuring FGT-B

Use the same firmware version as FGT-A and set the FortiGate unit to factory reset.
Go to **System > Dashboard > Status** and under "System Information" widget, change the host name to **FGT-B**



From CLI set the following:

```
FGT-B # config system ha
   set hbdev "ha1" 50 "ha2" 100
   set override disable
   set priority 128
   set frup enable
      config frup-settings
         set active-interface "wan2"
         set active-switch-port 14
         set backup-interface "wan1"
      end
end
```

```
FGT-B $ sh sys ha
config system ha
    set hbdev "ha1" 50 "ha2" 100
    set override disable
    set frup enable
        config frup-settings
            set active-interface "wan2"
            set active-switch-port 14
            set backup-interface "wan1"
        end
end

FGT-B $
```

“config frup-settings” are specific to the each FortiGate unit and not synchronized.

5. Results

With these settings:

- Traffic initiating from the server (IP: 192.168.1.200) connected to port16 should flow through FGT-A (since port16 is the active-switch-port in FGT-A) using WAN1 connection (since WAN1 is the active-interface in FGT-A).

Run a sniffer on both FortiGate units and then run a ping to 4.2.2.2 from the server.

```
FGT-A # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
0.231160 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
0.231202 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
0.231209 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.198520 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
1.198555 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
1.222569 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
1.222589 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.222595 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.199916 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
2.199952 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
2.232998 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
2.233017 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.233023 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.201347 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
3.201385 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
3.235406 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
3.235425 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.235430 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply

18 packets received by filter
0 packets dropped by kernel

FGT-A #
```

```
FGT-B # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
```

- Traffic initiating from an internal host (IP: 192.168.1.114) connected to port14 should flow through FGT-B (since port14 is the active-switch-port in FGT-B) using WAN2 connection (since WAN2 is the active-interface in FGT-B).

Run a sniffer on both FortiGate units and then run a ping to 74.125.226.1from the internal host.

```
FGT-A # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
```

```
FGT-B $ diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
1.887458 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
1.887488 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.887492 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.898137 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
1.898153 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.898159 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.885644 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
2.885682 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.885687 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.896175 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
2.896194 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.896201 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.884046 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
3.884091 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.884096 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.894192 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
3.894213 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.894220 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply

18 packets received by filter
0 packets dropped by kernel

FGT-B $
```

Shutdown FGT-A:

Traffic from the server and the internal host should be handled by FGT-B via WAN1 and WAN2 interfaces respectively:

```
FGT-B # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
0.954086 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
0.954226 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
0.968696 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
0.968780 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
0.968796 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.166934 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
1.166960 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.166966 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.177525 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
1.177541 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.177547 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.955117 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
1.955259 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
1.987992 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
1.988084 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.988101 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.165320 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
2.165346 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.165352 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.175081 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
2.175098 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.175105 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.956439 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
2.956583 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
2.973142 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
2.973237 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.973255 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.163683 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
3.163709 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.163714 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.174329 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
3.174362 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.174369 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.957570 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
3.957711 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
3.979899 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
3.979990 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.980012 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply

38 packets received by filter
0 packets dropped by kernel

FGT-B #
```

Shut down FGT-B:

Traffic from the server and the internal host should be handled by FGT-A via WAN1 and WAN2 interfaces respectively, but here, traffic is via WAN1 only. This seems to be a bug.

```
FGT-A # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
0.261907 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
0.262057 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
0.274912 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
0.275012 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
0.275024 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
0.409365 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
0.409621 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
0.419215 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
0.419305 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
0.419316 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.263358 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
1.263490 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
1.288955 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
1.289043 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.289054 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.408009 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
1.408141 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
1.418599 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
1.418674 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.418686 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.264384 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
2.264524 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
2.290900 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
2.290990 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.291000 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.406188 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
2.406307 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
2.415919 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
2.415991 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.416003 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.266330 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
3.266478 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
3.283646 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
3.283736 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.283746 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.405403 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
3.405529 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
3.418540 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
3.418614 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.418627 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply

40 packets received by filter
0 packets dropped by kernel

FGT-A #
```

Router-1 down/ unplug WAN1 interfaces:

Traffic from internal host and server is handled by FGT-B via WAN2 interface. Therefore FGT-A is NOT handling any traffic. FGT-A should handle some traffic on its WAN2 (backup) interface.

```
FGT-B $ diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
1.470201 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
1.470253 wan2 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
1.470258 eth0 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
1.483630 wan2 in 4.2.2.2 -> 172.20.120.130: icmp: echo reply
1.483649 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.483654 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.562495 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
1.562516 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.562520 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
1.572924 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
1.572941 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.572946 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.471997 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
2.472045 wan2 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
2.472050 eth0 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
2.485300 wan2 in 4.2.2.2 -> 172.20.120.130: icmp: echo reply
2.485318 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.485323 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.561119 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
2.561155 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.561160 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
2.571299 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
2.571317 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.571322 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.473663 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
3.473713 wan2 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
3.473718 eth0 out 172.20.120.130 -> 4.2.2.2: icmp: echo request
3.486935 wan2 in 4.2.2.2 -> 172.20.120.130: icmp: echo reply
3.486953 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.486959 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
3.559156 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
3.559180 wan2 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.559185 eth0 out 172.20.120.130 -> 74.125.226.1: icmp: echo request
3.568594 wan2 in 74.125.226.1 -> 172.20.120.130: icmp: echo reply
3.568610 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
3.568615 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply

36 packets received by filter
0 packets dropped by kernel

FGT-B $
```

```
FGT-A # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
```

Router-2 down/ unplug WAN2 interfaces:

Traffic from internal host and server is handled by FGT-A via WAN1 interface. Therefore FGT-B is NOT handling any traffic. FGT-B should handle some traffic on its WAN1 (backup) interface.

```
FGT-A # diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
0.974279 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
0.974410 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
0.989862 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
0.989949 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
0.989960 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.447625 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
1.447756 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
1.458029 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
1.458121 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.458132 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
1.976310 internal in 192.168.1.200 -> 4.2.2.2: icmp: echo request
1.976446 wan1 out 172.20.120.123 -> 4.2.2.2: icmp: echo request
1.990014 wan1 in 4.2.2.2 -> 172.20.120.123: icmp: echo reply
1.990095 internal out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
1.990106 eth1 out 4.2.2.2 -> 192.168.1.200: icmp: echo reply
2.447082 internal in 192.168.1.114 -> 74.125.226.1: icmp: echo request
2.447220 wan1 out 172.20.120.123 -> 74.125.226.1: icmp: echo request
2.457040 wan1 in 74.125.226.1 -> 172.20.120.123: icmp: echo reply
2.457126 internal out 74.125.226.1 -> 192.168.1.114: icmp: echo reply
2.457137 eth1 out 74.125.226.1 -> 192.168.1.114: icmp: echo reply

20 packets received by filter
0 packets dropped by kernel

FGT-A #
```

```
FGT-B $ diagnose sniffer packet any 'icmp' 4
interfaces=[any]
filters=[icmp]
```

Also to mention that after a fail over, the fall back is not achieved, and it requires a reboot on both FortiGate units. There are some bugs open regarding this behavior.